

### REMARKS

Claims 38-72 were pending at the time of examination. The applicants respectfully request reconsideration based on the foregoing amendments and these remarks.

#### **Claim Rejections – 35 U.S.C. § 102**

Claims 38 and 60 were rejected under 35 U.S.C § 102(e) as being anticipated by Czajkowski G. et al. "Internet Servers, Safe-Language Extensions, and Structured Resource Control," Technology of Object-Oriented Languages and Systems, 1999, Proceedings of Nancy, France 7-10 June 1999, Los Alamitos, CA, USA, IEEE Comput. Soc., Us, 7 June 1999, pages 295-304 (hereinafter "Czajkowski").

Czajkowski describes as system for alleviating "some problems related to resource management in server environments," (see Czajkowski abstract) and in particular to ascertain that a request that is received by a server does not consume or monopolize an entire resource in the server. In order to address this problem, an abstract concept, the "resource account," is introduced. The resource account "explicitly encapsulates the rights to various computational resources" (Czajkowski, page 5, 4<sup>th</sup> paragraph). Resource accounts can be shared by multiple threads, if needed. Whenever an attempt is made to charge more resources than allowed by a given resource account, an associated resource overuse callback is raised (Czajkowski, page 5, 5<sup>th</sup> paragraph). Thus, in Czajkowski a predetermined set of resources is dedicated in advance to one or more threads, based on the size a programmer chooses for the resource account.

The applicants' invention, as defined in claim 38, refers to "each code downloaded to the computer system." Thus, it is clear that the applicants' invention is not intended to be used in a server environment. Furthermore, the applicants' method, as defined in claim 38, requires "updating the resource indicator when the related code changes its actual collective resource usage." Thus, rather than pre-assigning a particular size to a resource account, as is done in Czajkowski, the method in the applicants' invention discovers when the related code changes its actual collective resource usage, and updates the resource indicator at that point. Thus, not only are the two methods different, but the applicants' method is more dynamic in nature than the method presented in Czajkowski, which uses a preset account size. For at least these reasons, it is submitted that the anticipation rejection of claim 38 is unsupported by the cited art and should be withdrawn. Furthermore, the applicants would also like to inform the Examiner that the Czajkowski reference cannot be cited under 35 U.S.C § 102(e), as Czajkowski is not a patent application publication or a granted patent.

Claim 60 is a *Beauregard* claim corresponding to claim 38. For reasons substantially similar to those set forth above with respect to claim 38, the applicants respectfully contend that the rejection of claim 60 is unsupported by the cited art and should be withdrawn.

#### Claim Rejections – 35 U.S.C. § 103

Claims 38-46, 50-58, 60-63, and 66-71 were rejected under 35 U.S.C § 103(a) as being unpatentable over U.S. Patent No. 5,838,968 to Culbert (hereinafter “Culbert”) in view of U.S. Patent No. 6,430,570 to Judge et al. (hereinafter “Judge”). Claims 47-49 and 64-65 were rejected under 35 U.S.C § 103(a) as being unpatentable over Culbert in view of Judge as applied to claims 38, 45, 60 and 63, further in view of U.S. Patent No. 6,182,022 to Mayle et al. (hereinafter “Mayle”). Claims 59 and 72 were rejected under 35 U.S.C § 103(a) as being unpatentable over Culbert in view of Judge as applied to claims 38 and 60, further in view of Applicant’s admitted prior art. The applicant respectfully traverses these rejections.

Claim 38, requires “for each code downloaded to the computer system, associating a resource indicator with all threads that are executed directly by the downloaded code and all threads that are initiated by the downloaded code, wherein all of the threads that are executed directly by the downloaded code and all threads that are initiated by the downloaded code are defined as a set of related code.” That is, a resource indicator is associated with all the related threads executed directly or indirectly by a particular downloaded code. Claim 38 also requires “updating the resource indicator when the related code changes its actual collective resource usage of a particular resource so that the resource indicator only tracks actual resource usage of the related code.”

In other words, the resource indicator is updated so as to track the actual resource usage of only the related code, which is defined as the threads executed or initiated by the particular code downloaded to a computer system. Thus, the resource indicator tracks actual resource usage of only the threads executed or initiated by the downloaded code. This feature advantageously allows implementation of procedures with respect to each set of threads executed or initiated on behalf of each downloaded code when actual resource usage by such related threads exceeds a particular limit. For example, these related threads can be terminated together when the resource indicator which tracks changes in actual resource usage only of these threads indicates that they are exceeding their resource usage. Furthermore, the resource indicator is updated “when the related code changes its actual collective resource usage.”

In contrast, Culbert appears merely to teach tracking of resource usage for each task running on a computer system or device (see col. 8, lines 38-59). However, Culbert fails to teach

or suggest tracking resource usage for all threads that are executed or initiated for each code downloaded to a computer system with a single resource indicator, in the manner claimed. Accordingly, threads originating from a downloaded set of related code cannot be dealt with together when one or more of the threads is misbehaving, as can be practiced with embodiments of the present invention. Instead, Culbert teaches using a resource tracking variable to track resource usage of an entire system or device. The applicants agree with the Examiner's assertion that the data for this resource tracking variable is obtained in Culbert by asking the individual tasks about their resource usage, but respectfully disagrees with the assertion that "related code is considered inherently included in each of the task execution which consume resources." If the related code were included in the execution of each task that consumes resources, it would be sufficient to ask only a subset of tasks (such as a set of "master tasks") for their actual resource usage, rather than asking all tasks for actual their resource usage.

Furthermore, even if the resource tracking variable of Culbert were to be considered to be equivalent to the resource indicators for the sets of related code in the applicants' invention, claim 38 also requires that the resource indicator be updated "when the related code changes its actual collective resource usage." In Culbert, on the other hand, "The UpdateResourceMeasurement routine is activated by a timer on a periodic basis..." (see col. 8, lines 47-48) or in response to a query from a resource manager (see col. 10, lines 52-55). Thus, the updates in Culbert are either "periodic" or "passive" in nature, and do not occur as a result of a set of related code changing its actual collective resource usage, as claimed.

In addition, the tasks in Culbert have three classes; error intolerant, error-tolerant realtime, and non-realtime (see Culbert, col. 8, lines 19-20), and the "error intolerant tasks never have their resource utilization records updated with actual use" (see Culbert, col. 8, lines 58-59). Thus, since the actual resource usage of the error intolerant tasks is never taken into account in Culbert, there would be no update if the error intolerant tasks were to change their resource usage, as required by claim 38.

The teachings of Judge do not remedy the deficiencies of Culbert in anticipating or rendering the applicants' invention obvious. Judge discloses resource management in a Java system, but fails to teach or suggest tracking resource usage with a single resource indicator for all threads that are executed or initiated for each code downloaded to a computer system, in the manner claimed. In Judge, "If, during the execution of an application...the JVM. 22 runs out of memory, an OutOfMemoryError error is generated" (Judge, col. 8, lines 1-3). The applicants respectfully disagree with the Examiner's assertion that "It is obviously for an ordinary skill in the art to recognize that the memory usage has been tracked for the executed applications," and

submit that the global memory usage can be tracked without tracking the usage of individual applications. There is no suggestion in Judge as to how to track the memory usage of individual applications.

In order to establish a *prima facie* case of obviousness, the Examiner must show a motivation to combine Culbert and Judge. Nothing in either Culbert or Judge suggests a desire to combine the two patents. The fact that both Culbert and Judge relate to different ways of managing low-memory conditions in computer devices or systems is not sufficient as a motivation to combine the two documents. Furthermore, the Examiner needs to show a reasonable expectation of success, which the Examiner has failed to do since he has not shown how Judge could cure the above mentioned deficiencies of Culbert. Finally, the combination of the references must teach or suggest all the claim limitations. Even if it were possible to combine Culbert and Judge, the combination still would not teach, for example, the limitation of "updating the resource indicator when the related code changes its actual collective resource usage of a particular resource so that the resource indicator only tracks actual resource usage of the related code" as neither Culbert nor Judge features this type of resource indicators or does any updates in the manner described by the limitation. For at least these reasons, the rejection of claim 38 is unsupported by the art and should be withdrawn.

Claim 60 is a *Beauregard* claim corresponding to claim 38. For reasons substantially similar to those set forth above with respect to claim 38, the applicants respectfully contend that the rejection of claim 60 is unsupported by the cited art and should be withdrawn.

The Examiner's rejections of the dependent claims are also respectfully traversed. However, to expedite prosecution, all of these claims will not be argued separately. Claims 39-69 and each depend directly from independent claims 38 or 60 and, therefore, are respectfully submitted to be patentable over cited art for at least the reasons set forth above with respect to claims 38 and 60. Further, the dependent claims require additional elements that when considered in context of the claimed inventions further patentably distinguish the invention from the cited art. For example, claims 57 and 70 further require that "determining which threads are to be defined as the set of related code based on which threads are assigned to a same protection domain." Claims 58 and 71 further require "aborting the threads of the related code when their resource indicator exceeds a maximum level." Finally, claims 59 and 72 require that "the computer system is integrated with a set top box or a navigational system." The cited references fail to teach or suggest such limitations.

**Conclusion**

The applicants believe that all pending claims are allowable and respectfully request a Notice of Allowance for this application from the Examiner. Should the Examiner believe that a telephone conference would expedite the prosecution of this application, the undersigned can be reached at the telephone number set out below.

Respectfully submitted,  
BEYER WEAVER & THOMAS, LLP



Fredrik Mollborn  
Reg. No. 48,587

P.O. Box 70250  
Oakland, CA 94612-0250